American Journal of Engineering and Applied Sciences

Original Research Paper

Identification of the Presence of the "Swollen Shoot" Disease in Endemic Areas in Côte d'Ivoire Via Convolutional Neural Networks

[1]Coulibaly Mamadou, [1]Silue Kolo, [1]Konan Hyacinthe Kouassi and [1,2]Olivier Asseu

[1]Department of Computer Science, Ecole Supérieure Africaine des TIC (ESATIC), Abidjan, Côte d'Ivoire
[2]Department of Electrical and Electronic Engineering Training and Research, INP-HB, Yamoussoukro, Côte d'Ivoire

Article history
Received: 04-06-2023
Revised: 26-06-2023
Accepted: 03-07-2023

Corresponding Author:
Coulibaly Mamadou
Department of Computer Science, Ecole Supérieure Africaine des TIC (ESATIC), Abidjan, Côte d'Ivoire
Email: mamadou.coulibaly@esatic.edu.ci

Abstract: The detection of Swollen Shoot disease and its control is one of the major objectives of research related to sustainable cocoa farming in Côte d'Ivoire. To contain the epidemic, the Cocoa Coffee Council (CCC) in collaboration with the National Agency for Support to Rural Development (ANADER) is responsible for prospecting and delimiting the infected areas as well as for uprooting suspect cocoa plants. since there is currently no cure for this virus. However, this monitoring is done with the naked eye and mobilizes many human resources (planters and plant pathologists). This process is delicate and time-consuming, resulting in significant economic losses for both planters and Côte d'Ivoire. Convolutional Neural Networks (CNN) emerged from the study of the visual cortex of the brain. CNNs are particularly used in image processing and offer many applications related to precision agriculture. Over the past few years, thanks to the increase in computing power, and the amount of training data available, CNNs have been capable of superhuman performance on complex visual tasks. They are at the heart of automatic image and video classification systems. The objective of the work presented in this article is to establish a collaborative solution between CNN-based image processing and plant pathology. The solution will reduce the human labor time required by using algorithms to facilitate the identification of swollen shoot disease in a cocoa plantation. The use of images collected from a drone on cocoa plantations as input information, allowed our learning model, based on CNNs, to guide a new approach for automating Swollen diagnosis. Shoot with our model, we have achieved a level of accuracy of 98% based on the known symptoms.

Keywords: Convolutional Neural Networks, Drone, Automatic Classification, Phytopathology, Swollen Shoot

Introduction

Plant diseases seriously harm production in quality and quantity by destroying possible agricultural advantages (Torres-Sánchez, 2014; Vasavi *et al*., 2022). It is crucial to contain plant diseases as soon as possible at the risk of seeing them spread through the plantation and destroy it. Swollen Shoot is a disease caused by a virus in cocoa plantations in West Africa where it causes significant production losses. Since 2003, Côte d'Ivoire has experienced a resurgence of the disease, the first outbreaks of which were discovered in 1946 and have spread to the entire cocoa production area to this day. Several techniques are used to identify the Swollen Shoot, on the one hand, it is recognizable in the plantation by observing the symptoms what are the swellings of the stems and/or the roots, the reddish or yellowish spots on the leaves, and the deformation of the pods, on the other hand it is detectable in the laboratory by molecular diagnosis (the ELISA test, Immunocapture-PCR and the Polymerase Chain Reaction (PCR)) (Oro, 2011; Oro *et al*., 2021). In view of the threat posed by the Swollen Shoot, the Ivorian government, through the Cocoa Coffee Council (CCC) has initiated a national program to combat the Swollen Shoot. This program aims to identify infected orchards, uproot them and then replant new cocoa plants. Nowadays, each plantation must be carefully examined regularly in order to identify and map all infected plots. This technique obviously requires the intervention of qualified people who have knowledge of the field and the process takes more time. Today, precision technology based on image processing is one of the driving forces of the modern agricultural revolution. The latter is based on data from satellites, drones, or even ordinary terrestrial cameras for crop monitoring. The use of this data makes it possible to identify potential problems (diseases) in specific areas of a plantation in order to treat them with targeted actions. This study will consist of setting up an efficient model for identifying the presence of Swollen Shoots on a plantation while minimizing human work in order to save considerable time. More specifically, we will collect images through a drone on cocoa orchards, develop and then train a model in order to identify Swollen Shoot disease, and make predictions of said disease.

Materials

For our data collection process, we opted for the DJI P4 Multispectral drone and its D-RTK2 base station (Fig. 1) because of their efficiency and simplicity in capturing data in photogrammetry/GIS work. The detection of visible symptoms is approached using an RGB color sensor while the other five sensors approach multispectral imaging for the study of potential early detection of diseased plants without visual symptoms.

Fig. 1: Datasheet of the material used

Technical Characteristics

- Aircraft: Take-off weight 1487 g, Max. 50 km/h (31 mph) (P-mode); 58 km/h (36 mph) (mode A)
- Camera: Six 1/2.9-inch CMOS, including one RGB sensor for visible light imaging and five monochrome sensors for multispectral imaging. Each sensor: 2.08 MP effective pixels (total number of pixels: 2.12 MP). Photo formats JPEG, TIFF
- Autonomy: 27 min
- GNSS: GPS + GLONASS + Galileo
- Radio control
- Intelligent flight battery (PH4-5 870 mAh-5.2V)
- Mapping functions: Ground resolution (GSD) (H/18.9) cm/pixel, where H is the altitude of the device relative to the mapped area (unit: m)

Methods

Observations via drone were made on the symptoms of the disease in each observation sub-plot. At the scale of the plantation, given the heterogeneity of plot areas, a diagonal experimental device was set up (Fig. 2).

Study Area

The choice of localities and prospected plantations was made using the survey technique. This technique consisted, in each region (Table 1), of drawing up, with the assistance of the supervisory structures, the list of localities because of their degraded sanitary condition and their easy access (Fig. 3). And in each of these localities, we chose two plots, one showing all the characteristic symptoms of Swollen Shoot disease (leaf, trunk, and pod) and the other being healthy.

Convolutional Neural Networks

The goal of CNN is to learn higher-order features in data through convolutions. Convolution is a simple operation on a table of numbers, matrix, or image to yield a transformation or derive key features from it. These networks are well suited for recognizing objects in images and they consistently rank high in image classification competitions. CNN architecture makes the implicit assumption that the input is image-like, which allows us to encode certain properties in the architecture. Convolutions capture translation invariance (i.e., filters are independent of the location).

This makes the transfer function more efficient, greatly reducing the number of parameters and thus making the network easier to optimize and less dependent on data size. Our problem is to determine whether the cocoa tree is healthy or diseased.

Table 1: List of plantations prospected by locality

Localities	Villages	Areas (ha)	Plots ages	Status
Abengourou	Amélékia	2,95	1995	Diseased
Abengourou	Amélékia	0,6	2013	Healthy
Daloa	Nioubouoa	1,52	2004	Diseased
Daloa	Bla	2,52	2016	Healthy
Aboisso	Ayamé	1,02	2016	Diseased
Sinfra	Proniani	3	2004	Healthy
Sinfra	Bétésso	7	1992	Diseased

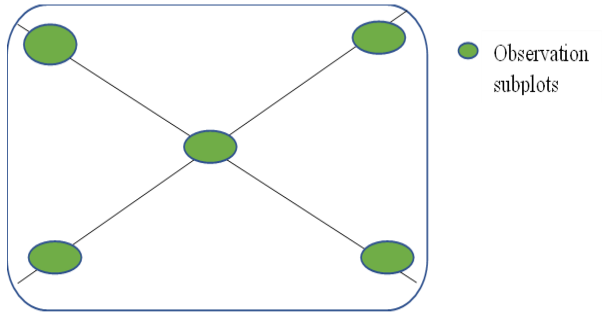

Fig. 2: Experimental device used for the observation of plantations

In contrast to regular neural networks, neurons in a CNN's layers are arranged in several dimensions, such as channels, width, height, and in the simplest 2D case, the number of filters. A convolutional neural network, like an MLP, consists of a series of layers, each transforming the activations or outputs of the previous layer by a different differentiable function. Several such layers are used in CNNs and are described in the next section. However, the most common building blocks encountered in most CNN architectures are convolutional layers, pooling layers, and fully connected layers. In essence, these layers are similar to feature extraction, dimensionality reduction, and classification layers respectively.

Before we continue with a top-level view of the exclusive layers, we pause a piece on the convolution layer. Essentially, a convolution layer makes use of a convolutional kernel as a clear-out for the entry. Usually, there are lots of such filters.

During an ahead pass, a clear-out slides throughout the entered quantity and computes the activation map of the clear-out at that factor via way of means of computing the pointwise product of every price and including those to acquire the activation on the factor. Such a sliding clear-out is clearly carried out via way of means of convolution and, as that is a linear operator, it may be written as a dot-product for green implementation.

Intuitively, because of this whilst educating this sort of CNN, the community will study filters that seize a few types of visible facts which include an edge, orientation, and eventually, in a better layer of the community, whole patterns. In every such convolution layer, we've got a whole set of such filters, every off with a view to produce a separate activation map. These activation maps are stacked to attain the output map or activation extent of this layer.

Convolutions

Mathematically, the convolution operation ($x * w$) (a) of the functions x and w is defined in all dimensions as:

$$(x * w)(a) = \int_{\infty}^{\infty} x(t) w(a - t) da \tag{1}$$

where, a is in R^n for all $n \geq 1$and the integral is replaced by its higher dimensional numerical variant. To understand the idea behind convolution, it is interesting to take the Gaussian function $w(a) = \exp(-x^2)$ as an example. If we take an image with the camera and shake the camera a little, the blurred image will be the real image x integrated with the Gaussian function w.

In convolutional neural network terms, x is called the input, w is called the filter or kernel and the output is often called the activation or feature map.

Note that we have modeled the input and kernel in (5.1). Due to the discrete nature of the image sensor, this will not happen in practice and it is more realistic to assume that the parameter t is discrete. If we assume this is the case, then we can define discrete convolution:

$$(x * w)(a) = \sum_a x(t) w(t - a) \tag{2}$$

where a span of all values in space can be in any dimension. In deep learning, x is usually a multidimensional data array and the kernel w consists of learnable parameters and usually has finite support, i.e., there are only a finite number of values a for which $w(a)$ is non-zero. This means that we can take (5.2) as a finite sum. The definition of (5.2) is dimension independent, but in medical imaging, we will mainly work with 2 or 3 dimensional convolutions:

$$(I * K)(i, j) = \sum_m \sum_n I(m,n) K(i - m, j - n) \tag{3}$$

$$(I * K)(i, j, K) = \sum_m \sum_n \sum_l I(m,n,l) K(i - m, j - n, k - l) \tag{4}$$

The convolutions (5.1) and (5.2) are commutative, which means that $I * K = K * I$ so that we can also write (5.3) as:

$$(I * K)(i, j) = \sum_m \sum_n I(i - m, j - n) K(m,n) \tag{5}$$

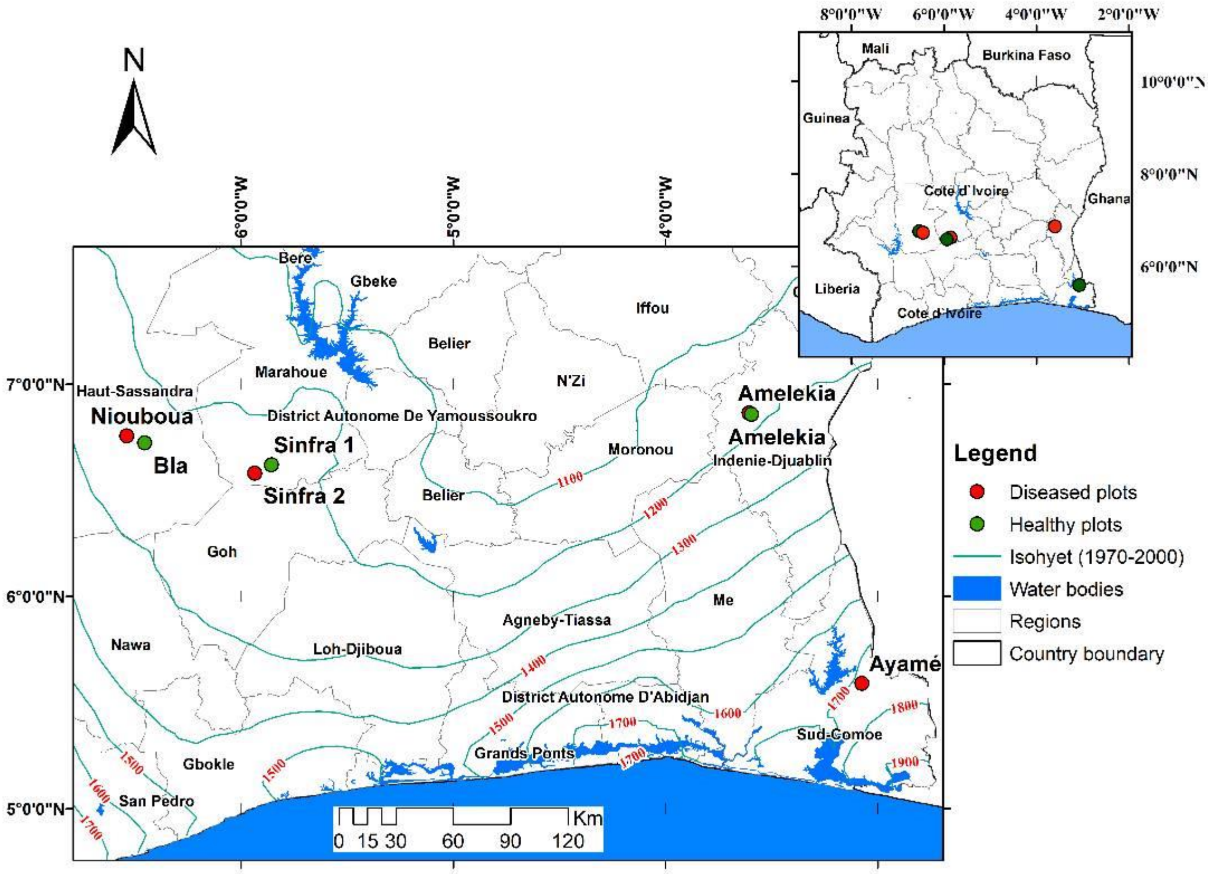

Fig. 3: Map of the study area

As K has finite support, this a priori infinite sum becomes finite. Some neural network libraries also implement an operation called cross-correlation, but from a deep learning perspective, these operations are equivalent, as one weight set can be directly translated into the other.

Nonlinearities

Nonlinearities are essential for neural network design: Without nonlinearities, a neural network would compute a linear function of its input, which is too restrictive. The choice of nonlinearity can have a large impact on the training speed of a neural network.

Sigmoid this nonlinearity is defined as:

$$\sigma(x) = \frac{1}{1+e^{-x}} \qquad x \in \mathbb{R} \tag{6}$$

Tanh the tanh nonlinearity is defined as:

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad x \in \mathbb{R} \tag{7}$$

ReLU this nonlinearity is defined as:

$$ReLu(x) = max(0,x) \quad x \in \mathbb{R} \tag{8}$$

Pooling Layers

Pooling consists of transforming a matrix into a smaller matrix while trying to keep its main characteristics. A pooling of size k transforms a matrix of size ($n \times p$) into a matrix of size k times smaller. The purpose of the clustering layer is to generate a summary statistic of its input and reduce the spatial size of the feature map (hopefully without losing necessary information). For this, the maximum clustering layer reports the maximum values in each rectangular neighborhood of each point (i, j) (or (i, j, k) for 3D data) of each input feature while the average clustering layer reports the mean values. The most common form of max pooling uses stride 2 with kernel size 2, partitioning the feature map space into a regular square or cube grid of side 2 and taking the maximum or average value on these cubes for each input feature.

DOI: 10.3844/ajeassp.2023.106.117

Although clustering operations are common building blocks of CNNs when the goal is to reduce the spatial size of the feature map, it should be noted that one can achieve the same goal using, for example, 3×3 convolutions with step 2 if working with 2D data. In this case, one can also simultaneously double the number of filters to reduce information loss while synthesizing higher-level functionality. For example, this down-sampling strategy is used in the ResNet architecture.

Fully Connected Layers

The fully connected layer with n input sizes and m output sizes is defined as follows. The output of the layer is determined by the following parameters: Weight matrix $W \in M_{m, n}(R)$ with m rows and n columns and polarization vector $b \in R^m$. Given the input vector $x \in R^n$, the output of the FC layer is fully connected to the activation function f defined as:

$$FC = f\left(Wx + b\right) \in R^m \tag{9}$$

In the formula above, Wx is the matrix product and the function f is applied component wise. Fully connected layers are used as final layers in classification problems, where a few (most often one or two) fully connected layers are attached on top of a CNN. For this, the CNN output is flattened and viewed as a single vector. Another example would be various autoencoder architectures, where FC layers are often attached to the latent code in both the encoder and decoder paths of the network. When working with a convolutional neural network it is helpful to realize that for a feature map with n channels, one can apply a convolution filter with kernel size 1 and m output channels, which would be equivalent to applying the same fully connected layer with m outputs to each point in the feature map.

CNN Architectures for Classification

The LeNet-5 architecture is probably the best-known CNN architecture (LeCun *et al*., 1989; 1998). It was created in 1998 and is widely used for handwritten digit recognition (MNIST). In recent years CNN based approaches, CNNs have become dominant in image classification, object localization, and image segmentation tasks. This popularity can be attributed to two major factors: Availability of computational resources (mostly GPUs) and data, on the one hand, and improvements in CNN architectures, on the other. Today CNN architectures have been developed that are quite successful in the tasks of image classification, object localization, and image/instance segmentation. Below we will discuss a few noteworthy CNN architectures for image classification problems. A neural network needs to have enough expressive power, depending on the task, to perform well. A naive approach to increasing the capacity is increasing the number of filters in convolutional layers and the depth of the network. This approach was taken in the AlexNet (Krizhevsky and Hinton, 2012) architecture, which was the first architecture that popularized CNNs in computer vision by winning the ImageNet ILSVRC challenge (Russakovsky *et al.*, 2015) in 2012. It featured 5 convolutional layers and only feed-forward connections with a total of 60M trainable parameters. Shortly after it was outperformed by:

- ZF Net (Zeiler and Fergus, 2014) in 2013
- GoogLeNet (Szegedy *et al.*, 2015), winning solution of the 2014 edition challenge
- VGG16/VGG19 (Simonyan and Zisserman, 2014), However, what marks the second big step forward in this project is that the overall performance outperforms the internet alone and becomes conceptually simpler

VGG19 has 19 feed-forward trainable layers of which 16 are convolutional and are based on 3×3 convolutions with stride 1 and ReLU enabled. The convolutional layers are grouped into 2 blocks of 2 layers for the first convolutional block and into 3 blocks of 4 layers for the last convolutional block. Max pooling is performed between convolutional blocks and the number of features in the convolutional blocks doubles after each max pooling operation. An important difference between AlexNet and VGG (as well as more modern architectures) is the effective receptive field size produced: AlexNet used an 11×11 filter in its original layer while VGG used a 3×3 filter stack and can easily prove that this is a more efficient way to increase the receptive field size. Compared to VGG19, GoogLeNet has fewer trainable parameters (4 M for 22 classes vs 4 M for 22 classes). 140 M for 19 layers of VGG19) is due to the introduction of the so-called starter module, which differs from the standard preview model and helps to improve the efficiency of the parameters.

However, all the architectures above still largely rely on feedforward information flow similar to the original LeNet-5 (LeCun *et al*., 1989; 1998), while the benefits of these architectures mostly stem from their depth. Making the feedforward architectures even deeper leads to several challenges in addition to an increased number of parameters. The first obstacle is the problem of vanishing/exploding gradients (Hochreiter, 1991; Bengio *et al*., 1994; Lerman *et al*., 2011). This can be largely addressed by normalized weight initialization and intermediate normalization layers, such as Batch Normalization (Ioffe and Szegedy, 2015). Yet another obstacle is the performance degradation problem (Torres-Sánchez *et al*., 2014). As the depth of a feedforward neural network increases, both testing and training accuracies get saturated and degrade afterward. Such performance degradation is not explained by overfitting and indicates that such networks are generally harder to optimize. Alternative CNN architectures have been suggested to deal with these shortcomings. A common feature of such architectures is the use of skip connections, which carry over information directly from earlier layers into the later layers without passing through intermediate convolutional layers. This, supposedly, helps, in general, to prevent information from "washing out". This general idea has been implemented in a number of ways.

System and Data Analysis

The system we propose is composed of four main steps:

- Step of pre-processing
- Step of learning
- Step of evaluation
- Step of prediction
- See Fig. 4

First, explore to identify a set of possible leads to constitute the input data of our system. The performance of the algorithm is dependent on the quality of the input data, it is therefore necessary to proceed to the cleaning of these because they constitute the training data. The cleaning step provides consistent data, with no outliers or missing values.

Data Preprocessing

If we consider two images that represent the same physical object, then they are linked together by a transformation so that it is possible to switch from one image to another. However, the exact characteristics of this transformation are still unknown to us. The problem is determining a transformation that can be transferred from one image to another, thus corresponding to a classical problem in computer vision, called image matching. Image matching involves finding elements common to two images called patterns represented by small images, known Patterns. Pattern matching task aims to find patterns in an image. The advantage of data preprocessing is that it makes it easier to understand and more suitable for model training.

Model matching is performed with filters using the cross-correlation operator, denoted ⊗. This operator transforms the image representing the matrix X into a new image $Y = H \otimes X$ as follows:

$$y_{i,j} = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H_{u,v} X_{i+u,j+v} \tag{10}$$

In this context, H is a small image representing the pattern found. Specifically, this operation is equivalent to dragging H over image X, to multiply the stacked pixels and sum these products.

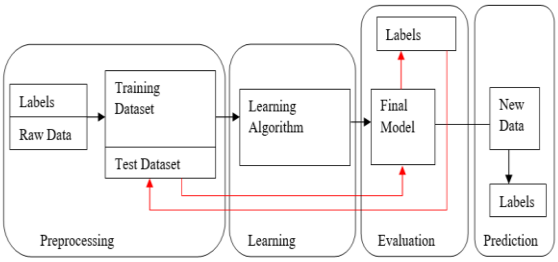

Fig. 4: Modeling our system

Fig. 5: Features defined by squares

Thus, pattern matching involves computing the correlation between image X and a filter whose kernel H represents the pattern one wants to find in X. In fact, this technique is not very practical. On the one hand, to be able to identify the pattern, you must find the area of the image to be recognized manually before performing the pattern matching. On the other hand, the cross-correlation operator is very sensitive to these variations. So, we need to find a way to get a more general model. In other words, it is necessary to formalize the properties of the feature elements of an image class. This is where the concept of an image's character comes in.

In computer vision, the term "features" refers to characteristic areas of a digital image. These areas can correspond to contours, points, or regions of interest. Each detected feature is associated with a vector, known as a descriptor (feature descriptor or feature vector), which, as the name implies, describes the region involved. Solving the image-matching problem is then performed in two steps (Fig. 5):

a) Detect and describe features in each image
b) Find matching feature pair in both images (feature match)

Image matching algorithms study the features of images, so the quality of the results depends (among other things) on how relevant the detected features are. In this sense, the first step is fundamental and should in no case be skipped.

The wrong selection of features can lead to some difficulties in the matching step:

- Problem 1: Two images do not have the same characteristics even though they represent the same object in different ways
- Problem 2: These two pictures have similar characteristics, but it is very difficult to find the matching pair

These two problems make matching impossible and therefore must be predicted from the first step when detecting and characterizing features. Which brings us to the next question:

- What characteristics should be selected?

These two problems make matching impossible and therefore must be predicted from the first step when detecting and characterizing features. Which brings us to the next question:

- Which features to choose from? Reproducible: A feature must be found in images that represent the same object despite geometrical and photometric differences. Therefore, a property must exhibit properties that are invariant to these transformations
- Special: A feature must be unique and clear enough in an image to facilitate the right fit. It is the information contained in its descriptor that should highlight its specificity
- Local: An object must correspond to a sufficiently small area and it should be described only in terms of its neighborhood. This helps to avoid corresponding difficulties due to clogging

A gradient is defined as being the generalization of the derivative of a function of one dimension into a function *f* having several dimensions, or variables. Image gradients is a vector ∇I consisting of partial derivatives of the intensity function and is pixel-by-pixel:

$$\nabla I(x,y) = \left(\frac{\partial I(x,y)}{\partial x}, \frac{\partial I(x,y)}{\partial y}\right) \quad (10)$$

The partial derivative with respect to *x* (or y) allows us to study the intensity variations of the image in the direction of the horizontal (or coordinate) axis.

Network Learning

Once our model design is complete, we will execute and run the model using the training data to tune the values of its parameters, thus using a learning algorithm. Learning is said to be unsupervised when it only provides training data for an algorithm without the desired output. Furthermore, in addition to presenting the training data as input to the algorithm and the desired output, the learning is said to be supervised. In the case of unsupervised learning, the algorithm's task is to find interesting relationships between the data or to partition them (clustering) according to predefined similarity criteria. Unsupervised learning aims to reduce the size of observations in some cases. In supervised learning, if the desired output takes its value in a finite set of cardinals, then the task performed is classification.

Deep learning is defined as a neural network with many parameters and layers and is built according to one of four basic architectures: Unsupervised learning network, convolutional neural network, neural network recurrent and recursive neurons. Flow diagram 4 is used to represent neural networks. The adjective "deep," which qualifies learning or neural networks, finds its origin in one property of flow charts. Lerman *et al*. (2011) the idea of deep neural networks has evolved very rapidly over time. Twenty years ago, a network with more than two hidden layers was considered deep. This idea of network depth is over. Today, the number of layers of deep neural networks is in the hundreds. Another characteristic of a deep neural network is the type of layers that make up it. New layers such as convolutional layers are unique to deep neural networks (Lecun *et al.*, 1998; Zeiler and Fergus, 2014). In the past decade, deep learning has been a topic of particular interest in the field of artificial intelligence (Copeland, 2006). Several large companies have embarked on the production of deep learning technologies, including Google, Facebook, Microsoft, and Yahoo.

Algorithm 1: Backpropagation

1: Input: m pairs of i.i.d. supervised Examples $(\tilde{x}_i, x_i)_{1\le i\le m}$
2: Parameter: connection weights and biases of the neural network θ:= $(W_l, b_l)_{1<l<n+1}$
3: Output: derivative of the objective $J_m = \frac{1}{2m}\sum_{i=1}^{m} V\ f(\tilde{x}_i,\theta) - x_i V_2^2$ with respect to θ.

Begin

4: Feedforward to get a m-column matrix D_{n+1} whose i^{th} column represents $f(\tilde{x}_i,\theta) - x_i$ as well as the m-column activation matrices $(A_l)_{1<l<n+1}$ whose ith column is formed by the values computed at layer *l* from the input $\tilde{x}_i$ according to (2)
5: **for** *l* ← n + 1 **to** 1 **do**
6: Compute the derivatives

$$\frac{\partial J_m}{\partial b_l} = \frac{1}{m} D_l 1_{m\times 1}$$

$$\frac{\partial J_m}{\partial W_l} = \frac{1}{m} D_l \mathrm{A}_l^T$$

$$\frac{\partial J_m}{\partial A_l} = W_l^T D_l$$

where $1_{m\times 1}$ is the column vector of m ones.

7: Set 1_l to be a matrix filled with ones and of the same dimension as A*l*

$$D_{l\text{-}1} = \frac{\partial J_m}{\partial A_l} \odot \left(1_l - A_l \odot A_l\right)$$

8: **end for**

9: Order $\left(\frac{\partial J_m}{\partial W_l}, \frac{\partial J_m}{\partial b_l}\right)_{1\le l\le n+1}$ the same way as

$(W_l, b_l)_{1<l<n+1}$ in θ to form $\frac{\partial J_m}{\partial \theta}$

End

Algorithm 2: Stochastic gradient descent

1: Input: Initial parameter θ_0 a training dataset and a distinct validation set $\left(\tilde{x}_i, x_i\right)_{1\le i\le N_1}$

2: Parameter: Learning rate *a*, iteration number *T*, observation interval *S*, batch size N_2

3: Output: a trained neural network θ_f.

Begin

4: Evaluate the current error on the validation set

$$\in := \frac{1}{N_1}\sum_{i=1}^{N_1} V\, f\left(\tilde{x}_i, \theta_0\right) - x_i V_2^2$$

5: **for** $t = 0$ **to** T-1 **do**

6: Draw N_2 pairs of supervised examples from the training dataset.

7: Calculate the stochastic gradient with Algorithm 1:

$$\frac{\partial}{\partial \theta} J_{N_2}\left(\theta_t\right)$$

8: Update the neural network

$$\theta_{t+1} = \theta_t - \alpha \frac{\partial}{\partial \theta} J_{N_2}\left(\theta_t\right)$$

Note that one may also want to set the learning rate in a layer-wise fashion, in which case α
is a positive diagonal matrix.

9: **if** mod $(t + 1, S) = 0$ **then**

10: Evaluate the current error on the validation set

$$\in := \frac{1}{N_1}\sum_{i=1}^{N_1} V\, f\left(\tilde{x}_i, \theta_{t+1}\right) - x_i V_2^2$$

11: **if** $\in > e$ **then**

12: $\in \leftarrow e$ and $\theta_f \leftarrow \theta_{t+1}$

13: end if

14: end if

15: end for

Network Assessment

Model validation is the process of understanding how it achieves a correct classification. There are two levels of validation: One at the level of model choice and the other at the level of model parameter selection. Indeed, several models are made and trained. The validation phase, therefore, consists of first selecting the model that best reflects reality and then adjusting the parameters again. This is done through testing. It is usual to provide a database dedicated to the tests in order not to bias the results. To this end, several specialists suggest separating the data collected into two or three groups (bases): A training base, a validation base, and possibly a test base (Lerman *et al*., 2011).

Results

The confusion matrix is a table whose rows and columns represent the predictions and actual results for a model. Our study was conducted on 3,480 cocoa plants to obtain the following confusion matrix (Table 2).

Let's call "positive" the class corresponding to a healthy cocoa tree and "negative" the other class. If we detect the symptoms of Swollen shoot when there is one, we make a "positive" prediction which is correct, it is a real positive. If, on the other hand, this prediction is incorrect, it is a false positive. And so on. False positives are also sometimes called "type I error" and false negative "type II error".

Recall, or sensitivity, therefore define recall ("recall" in English) or "sensitivity" in English, as the true positive rate, which measures how often we classify the input record as positive, this classification is proven correct. Here`s our model's ability to detect the presence of budding disease on images of cocoa and real people:

$$Recall = \frac{TP}{TP + FN} \quad (12)$$

Therefore, we will also be interested in accuracy (or positive predictive value), the degree to which repeated measurements under identical conditions give us the same results is called precision in scientific and statistical contexts. It is the percentage of correctly predicted points out of positively predicted points:

$$precision = \frac{TP}{TP + FP} \quad (13)$$

DOI: 10.3844/ajeassp.2023.106.117

Table 2: Confusion matrix

	Healthy pictures	Infected images	Total
Symptoms detected	72	1760	1832
No symptoms	1648	0	1648
Total	1720	1760	3480

Table 3: Architecture of model

Layer (type)	Output shape	Param #
Conv2d_4 (Conv2D)	(None, 254, 254, 16)	448
Max_pooling2d_3 (MaxPooling 2D)	(None, 127, 127, 16)	0
Conv2d_5 (Conv2D)	(None, 62, 62, 32)	0
Max_pooling2d_4 (MaxPooling 2D)	(None, 125, 125, 32)	4640
Conv2d_6 (Conv2D)		
Max_pooling2d_5	(None, 60, 60, 16)	4624
MaxPooling 2D)	(None, 30, 30, 16)	0
Flatten_1 (Flatten)	(None, 14400)	0
Dense_2 (Dense)	(None, 256)	3686656
Dense_3 (Dense)	(None, 1)	257

We are also often interested in specificity, which quantifies the ability of the model to avoid false positives. A cocoa tree with budding disease can be dangerous to other trees around it, so our model must be very sensitive to this situation and its effects. We regularly need to weigh the trade-off between sensitivity and specificity:

$$specificity = \frac{TN}{FP + TN} \tag{14}$$

Recall = 96%
specificity = 98%
Precision = 100%

Classical CNN architectures stack a few convolutional layers (each followed by usually a ReLU layer), then another composite layer, then a few more convolution layers (+ReLU), then another composite layer, and so on. The image shrinks as it passes through the network, but it also deviates deeper and deeper (the number of feature maps increases), thanks to convolutional layers. At the top of the stack, we typically add a classical no-loop neural network, consisting of a few fully connected layers (+ReLU) and the last layer that generates the prediction. The architecture of our model (Table 3) consists of three layers of convolution of decreasing size, the ReLU activation function, and a Max Pooling filter applied after each convolution. The network ends with a fully connected (dense) layer, consisting of a dense hidden layer and a dense output layer, a dropout layer (regularization technique to minimize network overfitting), and a "Sigmoid" activation function to do the prediction at the output because the sigmoid gives a value between 0 and 1, a useful probability for determining the two labels (healthy and infected).

Discussion

The results of our model thus obtained are promising because we notice that the loss curve (Fig. 6) tends towards zero (0) while the precision (Fig. 7). Tends towards one (1).

Comparison of Model Performance

We would have liked to compare our results only with those relating to the classification of cocoa tree diseases, unfortunately, very few studies exist to date. Therefore, we will compare our results with the results of the classification of plant diseases.

Comparison of the model trained in this study for the identification of swollen shoot disease with the results reported in the literature, relating to research to detect diseases in plants using convolutional neural networks, this table (Table 4) revealed that our model can distinguish and classify diseased swollen shoots with high accuracy.

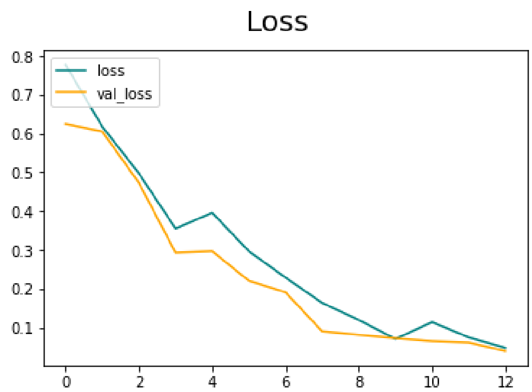

Fig. 6: Loss function

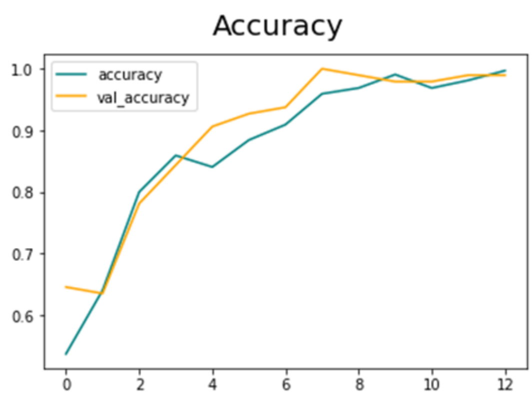

Fig. 7: Accuracy function

DOI: 10.3844/ajeassp.2023.106.117

Table 1: Comparison of model performance

	Culture	References	Accuracy %
1	Cacao	Coulibaly *et al.* (2020)	84
2	Potato	Bangari *et al.* (2022)	99
3	Cotton	Sharmila *et al.* (2023)	98
4	Rice	Gopi and Kondaveeti (2023); Kumar and Bhowmik (2023); Bharathiraja *et al*. (2023)	90-98
5	Tomato	Tantiborirak *et al*. (2023); Khalid *et al*. (2023)	87-99
6	Apple	Sahu *et al*. (2023); Sahu *et al*. (2023)	80-90
7	Sugarcane	Tanwar *et al.* (2023)	93
8	Coconut tree	Nivethitha *et al*. (2022)	85-97
9	Coffee	Pinto *et al*. (2021); Mridha *et al*. (2023)	96-99

Conclusion

In this article, we proposed a classification approach based on convolutional neural networks for the recognition of swollen shoot disease. We presented our preliminary results thus obtained. Experimental results indicate that our algorithm is feasible with a 98% recognition rate on all known symptoms. However, we recognize that the proposed method should be compared to another approach in order to assess its quality. Also, to improve it, we will further study the potential influence of other parameters so we will consider the identification of the range of host plants of the swollen shoot virus in cocoa plantations. This identification is an essential component of integrated pest management methods for this disease because it helps to reduce potential sources of infection and slow the progression of the disease in orchards.

Acknowledgment

We are grateful for the resources and platform provided by IMT Atlantique Brest and appreciate the efforts of the editorial team in reviewing and editing our work. We hope to contribute through this publication to help the Ivorian authorities to contain the swollen shoot disease.

Funding Information

The authors have not received any financial support or funding to report.

Author’s Contributions

Coulibaly Mamadou: Conceptualization, methodology, analysis, visualization, supervision and review.

Silue Kolo: Conceptualization, methodology and review.

Konan Hyacinthe Kouassi: Analysis supervision.

Olivier Asseu: Supervision.

Ethics

This article is original. The corresponding author confirms that all of the other authors have read and approved the manuscript and declares that there are no ethical issues that may arise after the publication.

References

Bangari, S., Rachana, P., Gupta, N., Sudi, P. S., & Baniya, K. K. (2022, February). A survey on disease detection of a potato leaf using CNN. In *2022 2nd International Conference on Artificial Intelligence and Smart Energy (ICAIS)* (pp. 144-149). IEEE. https://doi.org/10.1109/ICAIS53314.2022.9742963.

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157-166. https://doi.org/10.1109/72.279181

Bharathiraja, N., Pradeepa, K., Sheela, I. J., Sudhakar, G., Kumar, M. V., & Kaur, G. (2023, January). Prediction of Plant Leaf Diseases using Drone and Image Processing Techniques. In *2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT)* (pp. 1723-1727). IEEE. https://doi.org/10.1109/ICSSIT55814.2023.10061094

Copeland, M. (2016). What’s the Difference between Artificial Intelligence, Machine Learning, and Deep Learning. *Nvidia July*.

Gopi, S. C., & Kondaveeti, H. K. (2023, February). Transfer Learning for Rice Leaf Disease Detection. In *2023 3rd International Conference on Artificial Intelligence and Smart Energy (ICAIS)* (pp. 509-515). IEEE. https://doi.org/10.1109/ICAIS56108.2023.10073711

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. *Diploma*, *Technische Universität München*, *91*(1), 31.

Ioffe, S., & Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning* (pp. 448-456).

Khalid, A., Akbar, S., Hassan, S. A., Firdous, S., & Gull, S. (2023, February). Detection of tomato leaf disease using deep convolutional neural networks. In *2023 4th International Conference on Advancements in Computational Sciences (ICACS)* (pp. 1-6). IEEE. https://doi.org/10.1109/ICACS55311.2023.10089689

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, *25*.

Kumar, A., & Bhowmik, B. (2023, January). Rice cultivation and its disease classification in precision agriculture. In *2023 International Conference on Artificial Intelligence and Smart Communication (AISC)* (pp. 200-205). IEEE. https://doi.org/10.1109/AISC56616.2023.10085072

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541-551. https://doi.org/10.1162/neco.1989.1.4.541LeCun

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324. https://doi.org/10.1109/5.726791

Lerman, L., Markowitch, O., & Bontempi, G. (2011). Les systèmes de détection d'intrusion basés sur du machine learning. *M. Coulibaly et al. M. Coulibaly et al.*

Mridha, K., Tola, F. G., Khalil, I., Jakir, S. M. J., Wilfried, P. N., Priyok, M. A., & Shukla, M. (2023, April). Explainable Deep Learning for Coffee Leaf Disease Classification in Smart Agriculture: A Visual Approach. In *2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)* (pp. 1-8). IEEE. https://doi.org/10.1109/ICDCECE57866.2023.10151399

Nivethitha, T., Vijayalakshmi, P., Jaya, J., & Shriram, S. (2022, November). A Review on Coconut Tree and Plant Disease Detection using various Deep Learning and Convolutional Neural Network Models. In *2022 International Conference on Smart and Sustainable Technologies in Energy and Power Sectors (SSTEPS)* (pp. 130-135). IEEE. https://doi.org/10.1109/SSTEPS57475.2022.00042

Oro, F. Z., Lallie, H. D., Koné, N., Kouadio, J., & Diallo, H. A. (2021). Comparison of the prevalence of Cocoa Swollen shoot virus and the prevalence of Phytophthora sp in Petit-Bondoukou, South-West of Côte d'Ivoire. *International Journal of Environment, Agriculture and Biotechnology*, *5*(6). https://doi.org/10.22161/ijeab.56.33

Oro, Z. F. (2011). *Analyse des dynamiques spatiales et épidémiologie moléculaire de la maladie du swollen shoot du cacaoyer au Togo* (Doctoral dissertation, UM2).

Pinto, L. A., Mary, L., & Dass, S. (2021, August). The Real-Time Mobile Application for Identification of Diseases in Coffee Leaves using the CNN Model. In *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)* (pp. 1694-1700). IEEE. https://doi.org/10.1109/ICESC51422.2021.9532662

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, *115*, 211-252. https://doi.org/10.1007/s11263-015-0816-y

Sahu, K., Saraswat, T., Singhal, A., & Langer, G. (2023, April). CNN Based Disease Detection in Apple Leaf via Transfer Learning. In *2023 International Conference on Computational Intelligence, Communication Technology and Networking (CICTN)* (pp. 447-451). IEEE. https://doi.org/10.1109/CICTN57981.2023.10141259

Sharmila, S. K., Bhargavi, R., Anusha, R. Y., Anusha, K., & Divya, B. (2023, March). Cotton Leaf Disease Detection using Convolutional Neural Networks (CNN). In *2023 2nd International Conference on Electronics and Renewable Systems (ICEARS)* (pp. 1179-1185). IEEE. https://doi.org/10.1109/ICEARS56392.2023.10085551

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. https://doi.org/10.48550/arXiv.1409.1556

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1-9). https://doi.org/10.1109/CVPR.2015.7298594

Tantiborirak, S., Sukvichai, K., & Loraksa, C. (2023, January). Development of a Tomato Fruit Anomalies Detector for a Small Greenhouse Drone Application. In *2023 3rd International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP)* (pp. 41-44). IEEE. https://doi.org/10.1109/ICA-SYMP56348.2023.10044938

Tanwar, V., Lamba, S., Sharma, B., & Sharma, A. (2023, March). Red Rot Disease Prediction in Sugarcane Using the Deep Learning Approach. In *2023 2nd International Conference for Innovation in Technology (INOCON)* (pp. 1-5). IEEE. https://doi.org/10.1109/INOCON57975.2023.10101147

Torres-Sánchez, J., Peña, J. M., de Castro, A. I., & López-Granados, F. (2014). Multi-temporal mapping of the vegetation fraction in early-season wheat fields using images from UAV. *Computers and Electronics in Agriculture*, *103*, 104-113. https://doi.org/10.1016/j.compag.2014.02.009

Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13* (pp. 818-833). https://doi.org/10.1007/978-3-319-10590-1_53